



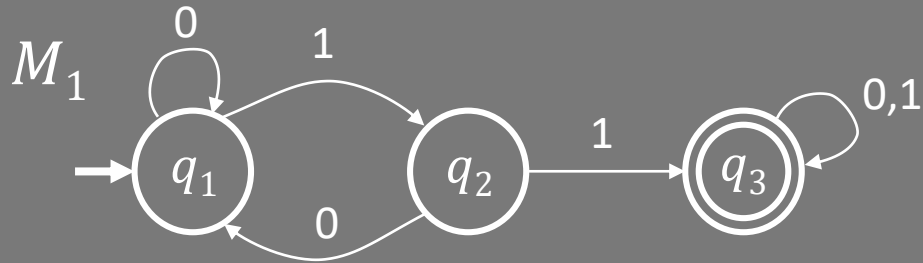
Department of Computer Science & Engineering

Presentation on Module - 1

**Prepared by
Dr. Pushpa R
Head & Professor
Dept. of CSE**



Let's begin: Finite Automata



States: $q_1 q_2 q_3$

Transitions: $\xrightarrow{1}$

Start state: $\rightarrow \bigcirc$

Accept states: $\bigcirc\bigcirc$

Input: finite string

Output: Accept or Reject

Computation process: Begin at start state, read input symbols, follow corresponding transitions, Accept if end with accept state, Reject if not.

Examples: 01101 \rightarrow Accept

00101 \rightarrow Reject

M_1 accepts exactly those strings in A where
 $A = \{w \mid w \text{ contains substring } 11\}$.

Say that A is the language of M_1 and that M_1 recognizes A and that $A = L(M_1)$.

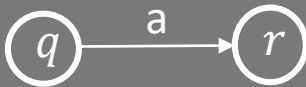
Finite Automata – Formal Definition

Defn: A finite automaton M is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$

Q finite set of states

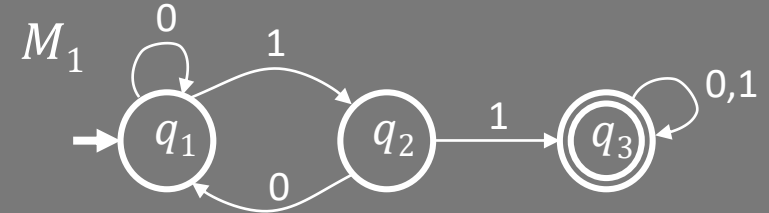
Σ finite set of alphabet symbols

δ transition function $\delta: Q \times \Sigma \rightarrow Q$

q_0 start state $\delta(q, a) = r$ means 

F set of accept states

Example:



$$M_1 = (Q, \Sigma, \delta, q_1, F)$$

$$Q = \{q_1, q_2, q_3\}$$

$$\Sigma = \{0, 1\}$$

$$F = \{q_3\}$$

$\delta =$		0	1
	q_1	q_1	q_2
	q_2	q_1	q_3
	q_3	q_3	q_3

Finite Automata – Computation

Strings and languages

- A string is a finite sequence of symbols in Σ
- A language is a set of strings (finite or infinite)
- The empty string ϵ is the string of length 0
- The empty language \emptyset is the set with no strings

Defn: M accepts string $w = w_1w_2 \dots w_n$ each $w_i \in \Sigma$ if there is a sequence of states $r_0, r_1, r_2, \dots, r_n \in Q$ where:

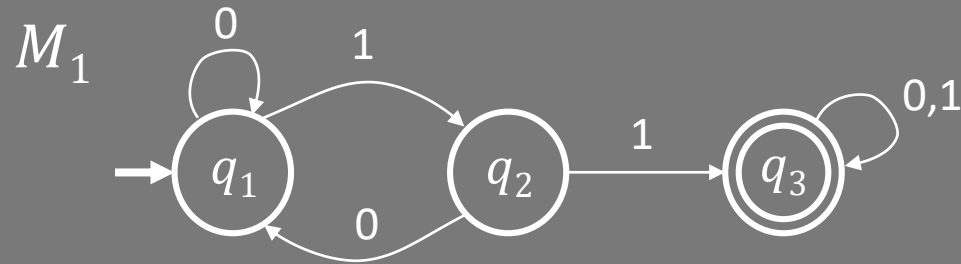
- $r_0 = q_0$
- $r_i = \delta(r_{i-1}, w_i)$ for $1 \leq i \leq n$
- $r_n \in F$

Recognizing languages

- $L(M) = \{w \mid M \text{ accepts } w\}$
- $L(M)$ is the language of M
- M recognizes $L(M)$

Defn: A language is regular if some finite automaton recognizes it.

Regular Languages – Examples



$L(M_1) = \{w \mid w \text{ contains substring } 11\} = A$

Therefore A is regular

More examples:

Let $B = \{w \mid w \text{ has an even number of 1s}\}$
 B is regular (make automaton for practice).

Let $C = \{w \mid w \text{ has equal numbers of 0s and 1s}\}$
 C is not regular (we will prove).

Goal: Understand the regular languages

Regular Expressions

Regular operations. Let A, B be languages:

- Union: $A \cup B = \{w \mid w \in A \text{ or } w \in B\}$
- Concatenation: $A \circ B = \{xy \mid x \in A \text{ and } y \in B\} = AB$
- Star: $A^* = \{x_1 \dots x_k \mid \text{each } x_i \in A \text{ for } k \geq 0\}$
Note: $\varepsilon \in A^*$ always

Example. Let $A = \{\text{good, bad}\}$ and $B = \{\text{boy, girl}\}$.

- $A \cup B = \{\text{good, bad, boy, girl}\}$
- $A \circ B = AB = \{\text{goodboy, goodgirl, badboy, badgirl}\}$
- $A^* = \{\varepsilon, \text{good, bad, goodgood, goodbad, badgood, badbad, goodgoodgood, goodgoodbad, ...}\}$

Regular expressions

- Built from Σ , members $\Sigma, \emptyset, \varepsilon$ [Atomic]
- By using $\cup, \circ, *$ [Composite]

Examples:

- $(0 \cup 1)^* = \Sigma^*$ gives all strings over Σ
- Σ^*1 gives all strings that end with 1
- $\Sigma^*11\Sigma^* =$ all strings that contain 11 $= L(M_1)$

Goal: Show finite automata equivalent to regular expressions

Closure Properties for Regular Languages

Theorem: If A_1, A_2 are regular languages, so is $A_1 \cup A_2$ (closure under \cup)

Proof: Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ recognize A_1

$M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ recognize A_2

Construct $M = (Q, \Sigma, \delta, q_0, F)$ recognizing $A_1 \cup A_2$

M should accept input w if either M_1 or M_2 accept w .

In the proof, if M_1 and M_2 are finite automata where M_1 has k_1 states and M_2 has k_2 states
Then how many states does M have?

- (a) $k_1 + k_2$
- (b) $(k_1)^2 + (k_2)^2$
- (c) $k_1 \times k_2$

Components of M :

$$Q = Q_1 \times Q_2 \\ = \{(q_1, q_2) \mid q_1 \in Q_1 \text{ and } q_2 \in Q_2\}$$

$$q_0 = (q_1, q_2)$$

$$\delta((q, r), a) = (\delta_1(q, a), \delta_2(r, a))$$

$$F = F_1 \times F_2 \quad \text{NO! [gives intersection]}$$

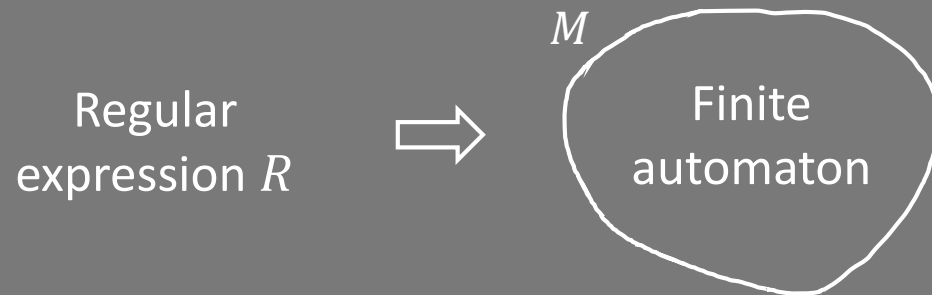
$$F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$$

Check-in 1.1

DFAs \rightarrow Regular Expressions

Recall Theorem: If R is a regular expression and $A = L(R)$ then A is regular

Proof: Conversion $R \rightarrow$ NFA $M \rightarrow$ DFA M'



Recall: we did $(a \cup ab)^*$ as an example

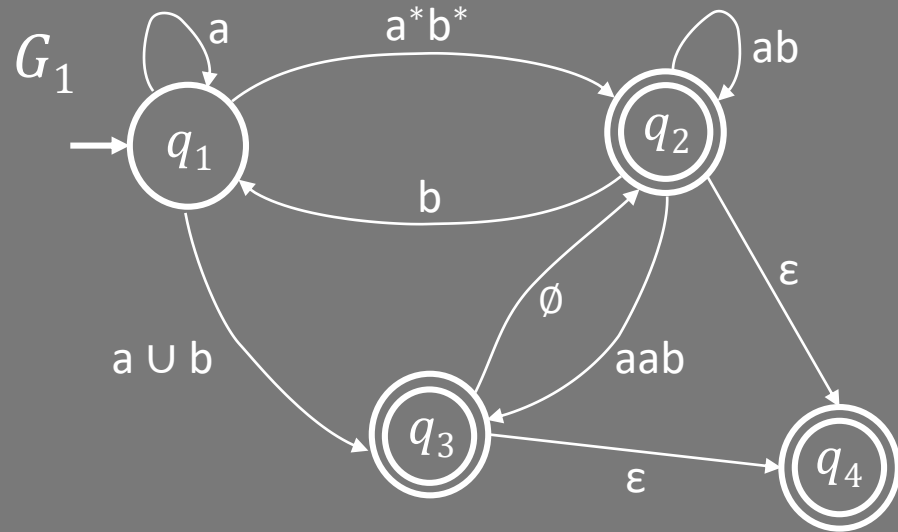
Today's Theorem: If A is regular then $A = L(R)$ for some regular expr R

Proof: Give conversion DFA $M \rightarrow R$

WAIT! Need new concept first.

Generalized NFA

Defn: A Generalized Nondeterministic Finite Automaton (GNFA) is similar to an NFA, but allows regular expressions as transition labels



For convenience we will assume:

- One accept state, separate from the start state
- One arrow from each state to each state, except
 - a) only exiting the start state
 - b) only entering the accept state

We can easily modify a GNFA to have this special form.

Context Free Grammars

$$G_1 \quad \left. \begin{array}{l} S \rightarrow 0S1 \\ S \rightarrow R \\ R \rightarrow \varepsilon \end{array} \right\} \text{(Substitution) Rules}$$

Rule: Variable \rightarrow string of variables and terminals

Variables: Symbols appearing on left-hand side of rule

Terminals: Symbols appearing only on right-hand side

Start Variable: Top left symbol

Grammars generate strings

1. Write down start variable
2. Replace any variable according to a rule
Repeat until only terminals remain
3. Result is the generated string
4. $L(G)$ is the language of all generated strings.

$$G_2 \quad \begin{array}{l} S \rightarrow RR \\ R \rightarrow 0R1 \end{array}$$

Check all of the strings that are in $L(G_2)$:

- (a) 001101
- (b) 000111
- (c) 1010
- (d) ε

$$L(G_1) = \{0^k 1^k \mid k \geq 0\}$$